REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 07-14-2006	**2. REPORT TYPE** Final performance report	**3. DATES COVERED** *(From - To)* 1/1/2003 to 12/30/2005

4. TITLE AND SUBTITLE
Canonical Probability Distributions for Model Building, Learning, and Inference

5a. CONTRACT NUMBER

5b. GRANT NUMBER
F49620-03-1-0187

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Marek J. Druzdzel, Ph.D.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Pittsburgh
School of Information Sciences
135 North Bellefield Avenue
Pittsburgh, PA 15260
Phone: 412-624-9432
Fax: 412-624-2788
Email: marek@sis.pitt.edu

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR
875 N Randolph St
Arlington VA 22203

AFRL-SR-AR-TR-06-0400

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The performed project focused on three major issues: (1) development and application of parametric conditional probability distributions, (2) improvements of stochastic sampling algorithms based on importance sampling, and (3) practical applications of our general purpose decision modeling environment to diagnosis of complex systems.

We have proposed a new class of parametric probability distributions, named Probabilistic Independence of Causal Interaction (pICI) models. We have shown that this class of models leads to improvements in learning of and inference in Bayesian networks. We have booked considerable advances in stochastic sampling algorithms for Bayesian networks based on importance sampling, preserved our leading role, and gained recognition of the community. Finally, we have developed a special module of **GeNIe** and **SMILE©**, the systems developed in the framework of our project, that supports diagnostic applications, and fielded the module in practical industrial settings.

15. SUBJECT TERMS
Bayesian networks, graphical models, uncertainty, decision making, parametric probability distributions, importance sampling, diagnosis.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Marek J. Druzdzel
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)* 412-624-9432

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Best Available Copy

Major Accomplishments

Major accomplishments of the project have been:

(1) definition and empirical evaluation of a new class of parametric conditional probability distribution models named Probabilistic Independence of Causal Interaction (pICI) models,

(2) a new stochastic sampling algorithm for Bayesian networks based on importance sampling, currently the state of the art in approximate algorithms for Bayesian networks, along with solid theoretical understanding of the most important heuristics used in importance sampling and extensions of importance sampling to continuous variables and probability distributions,

(3) software for Bayesian networks **GeNIe** and **SMILE**©, used by over 10,000 people world-wide, fielded at four major US corporations: General Electric, Boeing/Rockwell, Philips and Intel. In the framework of the current project, we have investigated, designed, and built two major modules of the software:

(a) diagnostic software module,

(b) learning and data mining software module,

In addition to these accomplishments, we have done the following successful research acknowledging AFOSR support:

(4) Proof of correctness of the Causal Ordering Algorithm

(5) Annealed MAP and A*-based MAP algorithms

(6) A Robust Independence Test for Constraint-Based Learning of Causal Structure

(7) An Efficient Sampling Algorithm for Influence Diagrams

We briefly summarize each of these accomplishments in the separate sections below.

Probabilistic Independence of Causal Interactions Models

We have conducted a thorough review of the existing canonical distributions and wrote a review paper that is under review by the *Knowledge Engineering Review* journal (Diez & Druzdzel, draft). Another document on this topic is a "comprehensive examination" paper written by a doctoral student funded by the grant, Adam Zagorecki. Mr. Zagorecki is about to defend his dissertation and we expect that this document will also grow to become a review paper.

We have studied the CAST model, popular in military applications. Our conversations with Dr. John Lemmer of the Air Force Rome Laboratories, should lead to a join paper critically reviewing the CAST model and comparing it to the *recursive Noisy-OR*, a causal interaction model developed by Dr. Lemmer and his collaborators at the AFRL.

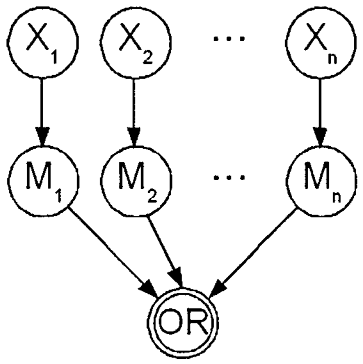
We progressed on an empirical study of existing models with respect to their conformance to the Noisy-MAX model (Zagorecki & Druzdzel, 2006), tested the two alternative parameterizations of the leaky Noisy-MAX gate with respect to their accuracy in probability elicitations (Zagorecki & Druzdzel, 2004).

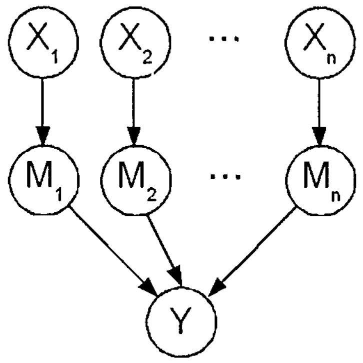
We have developed a new class of canonical models, and a useful instance of this class, the noisy average model (Zagorecki & Druzdzel, draft).

PICI Models

Formally, a BN is a compact representation of a joint probability distribution (JPD). It reduces the number of parameters required to specify the JPD by exploiting independencies among domain variables. These independencies are typically encoded in the graphical structure, in which nodes represent random variables and lack of arcs indicate probabilistic conditional independencies. The parameters are specified by means of local probability distributions associated with variables. In case of discrete variables (the focus of this paper), the local probability distributions are encoded in the form of prior probabilities over nodes that have no parents in the graph and conditional probability tables (CPTs) for all other nodes. Specifying a series of CPTs instead of the JPD typically reduces the number of required parameters significantly, depending on the structure of the graph. However, the number of parameters required to specify a CPT for a node grows exponentially in the number of its parents. Effectively, the size of CPTs is a major bottleneck in building or learning models, and in reasoning with them. For example, assuming that all variables are binary, a CPT of a variable with 10 parents requires the specification of 210 = 1,024 probability distributions. If another parent is introduced, the number of required distributions will grow to 2,048. This may be overwhelming for an expert if the distributions are elicited. If the distributions are learned from a small data set, there might not be enough cases to learn the distributions for all the different parent configurations in a node (Onisko, Druzdzel, & Wasyluk 2001).

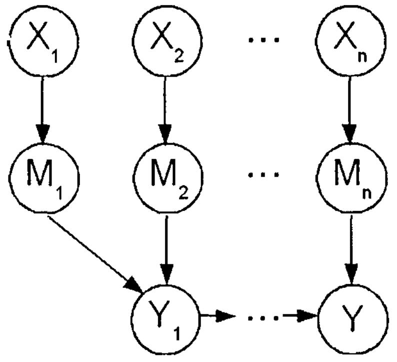
We introduced a new class of parametric models that require significantly fewer parameters to be specified than CPTs. The new models are a generalization of the class of Independence of Causal Interactions (ICI) models (Heckerman & Breese 1996) (they call it causal independence models), and its unique feature is that the combination function does not need to be deterministic. The combination function takes as input the values of the parent variables and produces a probability distribution over the values of the child variable. The most important property of the new class is that the combination functions are potentially decomposable, which leads to substantial advantages in inference. We have decided to call the newly proposed class pICI or *probabilistic ICI.* The following two figures show the difference between the ICI and the pICI classes:

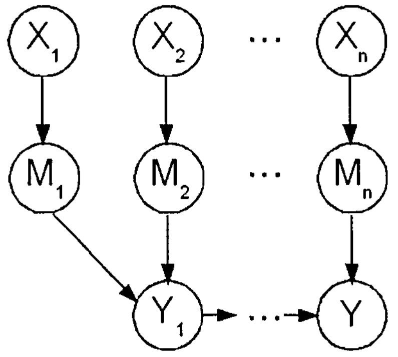


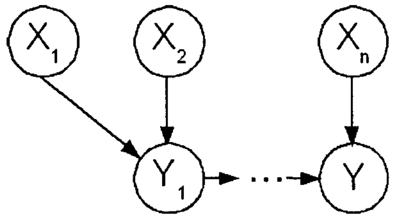


The ICI models (left) have the combination function at Y deterministic (in this picture, it is the deterministic OR function; other combination functions used in practice are: AND, XOR, MIN, MAX, and SUM). In the pICI models, we allow a probabilistic combination function and, hence, extend the class of problems that can be represented. A deterministic function can be viewed as a special case of a probabilistic function, one in which all probabilities are degenerate and are all zeros and ones.

The pICI distributions can be decomposed structurally, resulting in significant savings in parametrization. Here are two possible decompositions that we call ladder model, average model, and simple ladder model.

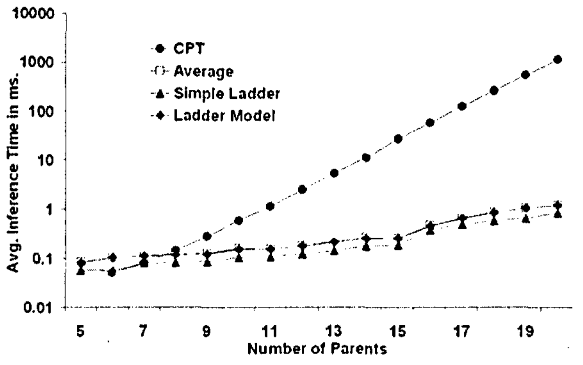


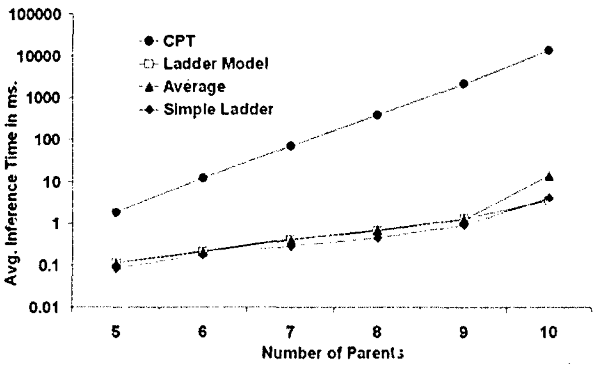




The pICI models have two main advantages over CPTs. The first advantage is that inference may be faster, because the decompositions result in smaller clique sizes in the joint tree algorithm (Zhang & Yan 1997). This becomes especially dramatic when the number of parents is large. The second advantage is that if the decompositions (rather than CPTs) are learned from a small data set, the resulting network is likely to be more faithful in representing the true underlying probability distribution. This is because a smaller number of parameters will prevent the decompositions from over fitting the data.

We compared empirically the speed of exact inference between CPTs and the new models, using the joint tree algorithm (Lauritzen & Spiegelhalter 1988). We were especially interested in how the new models scale up when the number of parents and states is large compared to CPTs. We used models with one child node and a varying number of parents ranging from 5 to 20. We added arcs between each pair of parents with a probability of 0.1. Because the arcs between parent nodes can influence the inference times, we repeated the procedure of generating arcs between the parents 100 times and took the average inference time for the 100 instances. The last parameter to fix is the number of states in the variables and we subsequently used 2, 3, 4, and 5 states for all the variables. Because of the computational complexity, not all experiments completed to the 20 parents. When there was not enough memory available to perform belief updating in case of the CPT network, we stopped the experiment. The results are presented in the figures below. We left out the results for 3 and 4 states, because the only difference is the exact point of intersection with the y-axis.



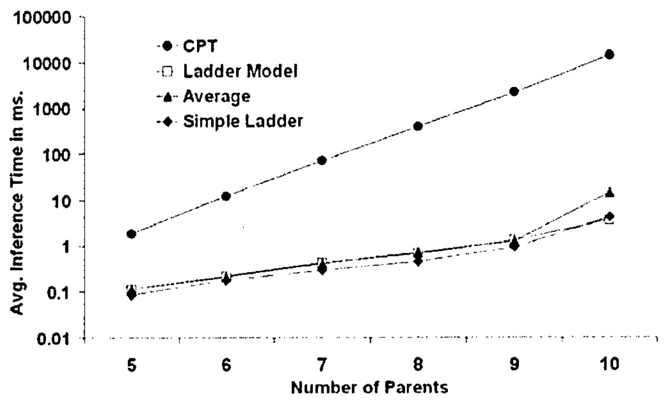


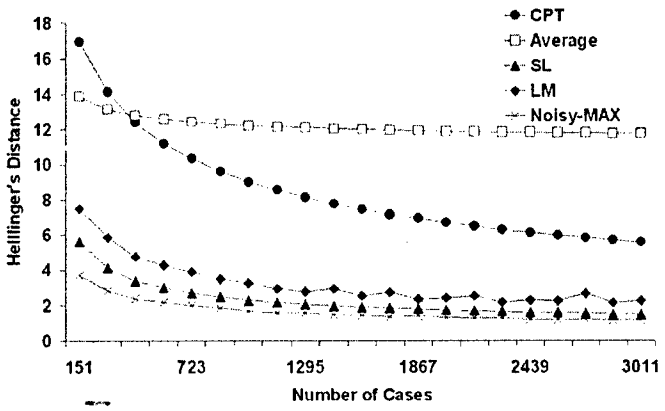
It is easy to notice that the decomposable models are significantly faster for a large number of parents, and the effect is even more dramatic when more states are used. The improvement in speed is logarithmic, as can be expected from theoretical considerations. Heckerman&Breese [1994] empirically showed that if decompositions are used in general BNs, it will speed-up inference significantly. The differences obtained are highly statistically significant.

To verify that the inference speed-up not only holds for families, but also for Bayesian networks in general, we selected the CPCS179 network [Middleton et al. 1991, Shwe et al. 1991] to perform an additional experiment. The idea was to decompose a family only if the number of parents was greater or equal to *n* (we tried *n* between 2 and 10) and measure the duration of 1,000 times belief updating. The results are displayed below. It is clear that when the value of *n* is low, the decompositions do not pay off. From *n* = 3 to *n* = 7 the difference in inference times is easily noticable. The reason why there was no inference speedup for n > 7 is that there were almost no nodes decomposed.

In the following experiment, we investigated empirically how well the decompositions can be learnt from small data sets. We selected gold standard families (a family is a child plus its parents) that had three or more parents from the following real-life networks (available at http://genie.sis.pitt.edu/): HAILFINDER [Edwards 1998], HEPAR II [Onisko, Druzdzel & Wasyluk 2001] and PATHFINDER [Heckerman, Horvitz & Nathwani 1992]. We generated a complete data set from each of the selected families. Now the idea is to randomize all the parameters and try to relearn the original 'gold standard' parameters from the generated data sets. To do the parameter learning we used the EM algorithm [Dempster, Laird & Rubin 1977] to relearn the parameters of the CPTs and decomposed models. We repeated this procedure 50 times for different data sets. The number of cases in the data sets ranged from 10% of the parameters in the CPT, to 200%. For example, if a node has 10 parameters, the number of cases used for learning ranged from 1 to 20. In learning, we assumed that the models are decomposable, i.e., that they can be decomposed according to the three decompositions pictured earlier. The main difference between the LM and Average model is that in the Average model the combination function is predefined, and in the LM model the combination function was learned. Note that the EM algorithm is especially useful here, because the decompositions will have hidden variables (e.g., the mechanism nodes). The EM algorithm is able to handle missing data. Our hypothesis was that the decompositions learn better than CPTs as long as the number of cases is low. We compared the original CPTs with the relearned CPTs, decompositions, and noisy-MAX using Hellinger's distance [Kokolakis & Nanopoulos 2001]. To account for the fact that a CPT is really a set of distributions, we defined a distance between two CPTs as the sum of distances between corresponding probability distributions in the CPT weighted by the prior probability of the parents of this distribution. This approach is justified by the fact that in general it is desired to have the distributions closer to each other when the parent configuration is more likely. If this is the case, the model will perform well for the majority of cases.

We decided to use Hellinger's distance because, unlike the Euclidean distance, it is more sensitive to differences in small probabilities, and it does not pose difficulties for zero probabilities as is the case for Kullback-Leibler divergence [Kullback & Leibler 1951]. We selected three nodes, one from each network, and show the results in the two figures below. It is clear that the CPT network performs poorly when the number of cases is low, but when the number of cases increases, it comes closer to the decompositions. In the end it will fit better, because the data is generated from CPTs. For node F5 from the PATHFINDER network the Average model provided a significantly worse fit than the other models. This means that the Average model did not reflect the underlying distribution well. For other distributions the Average model could provide a very good fit, while, for example, the noisy-MAX model performs poorly. Again, it is important to emphasize that the pICI models performed better for almost all the decomposed nodes as is shown in the next paragraph.

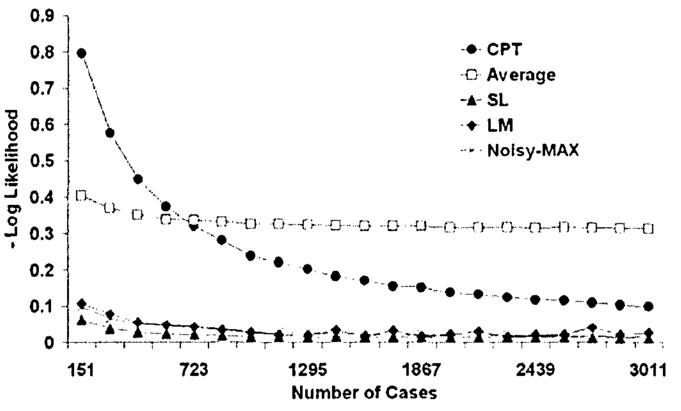


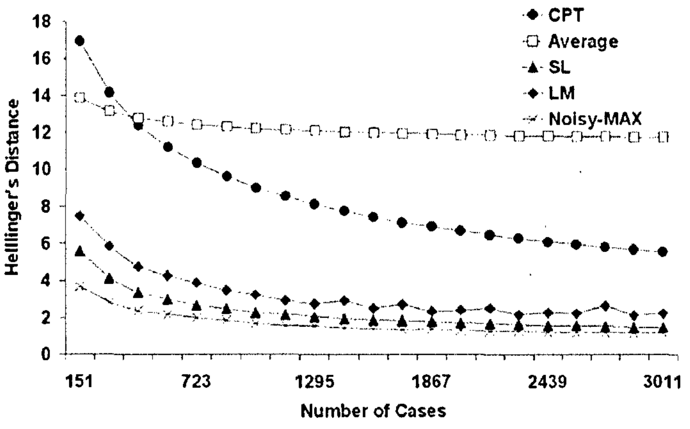


Similarly to the first experiment, our method should scale to larger CPTs and we can expect more dramatic results there.

There are no general a priori criteria to decide which model is better. Rather these models should be treated as complementary and if one provides a poor fit, there is probably another model with different characteristics that fits better. The last experiment in this series studies the problem of automatic selection of the decomposition that will fit best. in real-life the true underlying probability distribution is not known. Hence, we have to use the available data for selecting an appropriate decomposition. Our hypothesis, confirmed by the experiment, was that the likelihood function will provide a fast and easy way of guessing which decomposition will perform best.

We performed an experiment to test if it is possible to use the likelihood function of the data, to see which model fits the data best. The likelihood function is defined as p(d|ζ), where d stands for a data set and ζ for the set of parameters. The likelihood function is a conditional probability function considered a function of ζ with the data set fixed. It assigns the highest probability to the parameters that fit the data best. The results are surprisingly good in the sense that the results of the likelihood function are very similar to those obtained by using Hellinger's distance. We used cross-validation to verify if the likelihood function is suitable to select the best decomposition. The experimental setup was the following. We used the same families as in the first experiment, generated a data set from the gold standard model, and split it into a training and test set. We used the training set to learn the model and a test data set of the same size as the training set to calculate the likelihood function. The following figure shows Hellinger's distance (right) and the corresponding likelihood function (left) for node F5.





The shapes of the functions are essentially the same, showing that the likelihood function is a good predictor of model fit. This was true in general for all the learned models.

The newly introduced class of parametric models, the pICI models, relaxes some assumptions of causal independence models and that allow for modeling a wider variety of interactions than the ICI models. The pICI models have a probabilistic combination function that takes the values of the input variables and produces a probability distribution over the values of the output variable. We focused on a subset of the new class of models with decomposable combination functions. We showed the results of an empirical study that demonstrates that the decompositions lead to significantly faster inference. We also showed empirically that using these models for parameter learning with the EM algorithm from small data sets, results in networks that are closer to the true underlying distribution than what it would be with CPTs. Finally, we demonstrated that the likelihood function can be used to select the decomposition that fits the model best.

The pICI models are intended for usage in real-life networks when a child node has a large number of parents and, therefore, the number of parameters in its CPTs is prohibitively large. In practice, this happens quite often, as is observed in the Bayesian networks that we used in our experiments.

Stochastic sampling algorithms

One of the lines of the proposed work focused on stochastic sampling algorithms for Bayesian networks. Our laboratory has been the leader in importance sampling algorithms, currently the fastest and most accurate stochastic sampling algorithms. Our work and our results have been recognized by the community. We have received Honorable Mention in the *2005 IJCAII-JAIR Best Paper Prize* for the paper with Jian Cheng "AIS--BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks." The IJCAII-JAIR Best Paper Prize is awarded to an outstanding paper published in JAIR in the preceding five calendar years. For the 2005 competition, papers published between 2000 and 2005 were eligible. The primary funding source for that paper was our previous AFOSR grant.

In our most recent work, in the framework of the 2003-2005 grant, we have made further improvements on the importance sampling algorithms, increasing their speed and accuracy and developed theoretical insight into the power of the heuristics applied in the AIS-BN and the EPIS-BN algorithms. We have developed solid theoretical understanding of the concept of heavy tails in sampling distribution and heuristics that assure heavy tails and improve the accuracy of sampling (Yuan & Druzdzel, 2004, 2004a, draft). We have also extended our work to continuous probability distributions. While fine details can be found in our publications, this report summarizes the main results.

Our focus on stochastic sampling algorithms is motivated as follows. A system that is a combination of Bayesian networks and structural equation models, our long-term goal, needs to include algorithms that are flexible enough to work with both discrete (Bayesian networks) and continuous (structural equation models) variables. The algorithms have to accommodate arbitrary probability distributions and work with very large models. The only known classes of algorithms that will accommodate these requirements are stochastic sampling algorithms. In our work (starting with the previous AFOSR grants), we probed three directions: Latin hypercube sampling, quasi-Monte Carlo methods, and adaptive importance sampling. In some of the papers resulting from the previous grants we also acknowledged the current grant when the papers were published after 2002, as additional experiments or finishing touches on the papers were performed after the termination date of the previous grant.

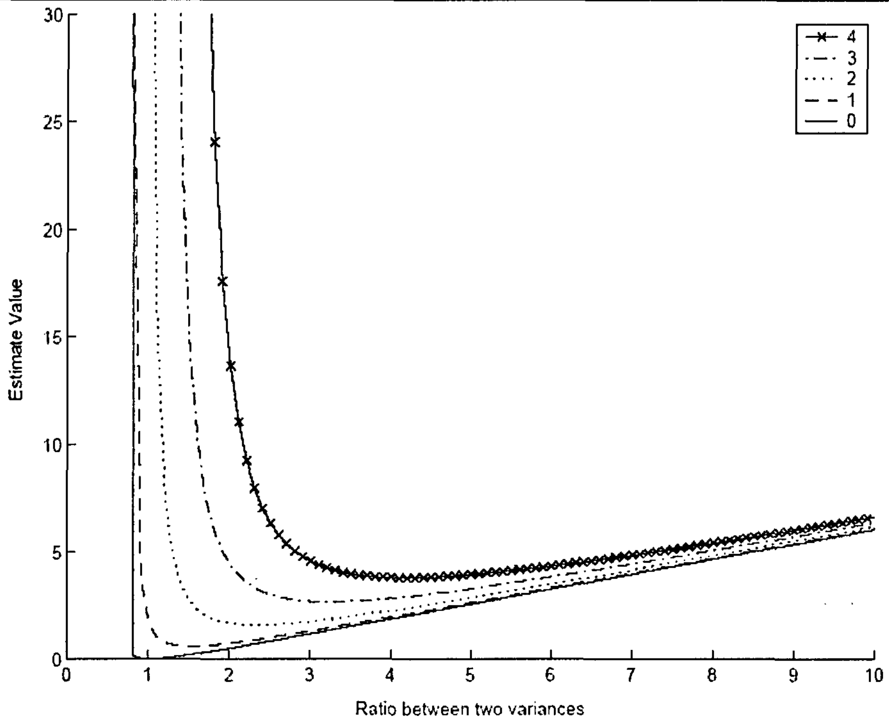
Our results published in a paper on the AIS-BN algorithm (this was the JAIR paper that has received honorable mention in the *2005 IJCAII-JAIR Best Paper Prize*) were excellent – on real, hard cases, when the probability of evidence is very low, the algorithm has beaten previous algorithms by two orders of magnitude in terms of its precision. In terms of computing time required to reach the same precision, the results were even better.

The EPIS-BN algorithm, the development of which started in the framework of the previous grant but has completed in the course of this grant, improves these phenomenal results even further. We tested it on several large real Bayesian networks and compared the results with the AIS-BN algorithm. The empirical results showed that the EPIS-BN algorithm provides a considerable improvement over the AIS-BN algorithm, especially in those cases that are hard for the latter.

Essentially, existing algorithms differ only in the methods that they use to obtain the importance functions I(X). The closer I(X) to the actual posterior distribution, the better the performance. A good importance function can lead importance sampling to yield good convergence results in an acceptable amount of time. It is well understood that we should focus on sampling in the areas where the value of the posterior distribution is relatively large [Andrieu 2003, Rubinstein 1981], and, hence, I(X) should concentrate its mass on the important parts of the posterior distribution p(X). However, unimportant areas should by no means be neglected. Several researchers pointed out that a good importance function should possess thicker tails than the actual posterior distribution [Geweke 1989, Mackay 1998]. However, the desirability of thick tails was much less understood. In our paper on this issue [Yuan & Druzdzel 2005, Yuan & Druzdzel, under review], we try to address the limitation and develop a better understanding for the requirement. First, we explain the basic assumptions behind importance sampling and their importance. We then study the properties of importance sampling and discuss what conditions an importance function should satisfy. After that, we specifically study the properties of importance sampling in the context of Bayesian networks, which leads to several theoretical insights into the desirability of thick tails. The insights not only shed light to the success of the AIS-BN algorithm, which relies on two heuristic methods to obtain an initial importance function: ε heuristic, replacing small probabilities in the conditional probability tables by a larger ε, and setting the probability distributions of the parents of evidence nodes to uniform, but also provide a common theoretical basis for several other successful heuristic methods.

The basis of our analysis is the work published by the PI in 1994 [Druzdzel 1994] on some properties of joint probability distributions. In that paper, I have postulated that the probability distribution over the probability of individual components of a joint probability distribution follows the log-normal distribution. I have supported this with a theoretical analysis based on the Central Limit Theorem and with several empirical studies.

We have derived a closed form solution for the variance of the estimator in importance sampling in Bayesian networks and have shown that we can look at any importance sampling algorithm for Bayesian networks as using one lognormal distribution for the importance function I(X) to compute the expectation of another lognormal distribution p(X). We have plotted the solution for the variance of the estimator in the figure below for various values of the ratio between the standard deviations of I(X) and the standard deviation of p(X).



There are several observations that can be made from this plot:

- Given the value of fraction, the variance is monotonically increasing as the ratio increases.

This observation is consistent with the well understood requirement that I(X) should concentrate its mass on the important parts of the posterior probability mass. The more I(X) misses the important parts of p(X), the worse importance sampling performs.

- Given the value of mean of I(X) and, hence, the value of the fraction, there is a minimum variance when the fraction takes a particular value, say *u*. As the fraction decreases from *u*, the variance increases quickly and suddenly goes to infinity. When the fraction increases from *u*, the variance also increases but much slower.

- As the fraction increases, the performance of I(X) with different mean differ less and less.

The above two observations clearly tell us that if we do not know the precise value of the fraction, i.e., we are not sure if I(X) covers the important parts of posterior probability distribution or not (We use the term cover to mean that the weight of one density is comparable to that of another density in a certain area.), we may want to make the tails of I(X) thicker in order to be on the safe side. We can see that in that case the variance of the estimator becomes larger, but not much larger.

- The *u* value increases as fraction increases, which means that the more I(X) misses the important parts of the posterior probability distribution, the thicker the tails of I(X) should be.

The four observations all provide strong support for thick tails. In practice, we usually have no clue about the real shape of posterior probability distribution. Even if we have a way of estimating it, our estimation may not be that precise. The main lesson from our analysis is that we want to avoid light tails and err on the thick tail side in order to be safe. One possible strategy is that we can start with an importance function with considerably thick tails and refine the tails as we gain more and more knowledge about it.

It can be shown that the above results hold not only for Bayesian networks but also for several well-known distributions, including normal distribution. Although generalizing the results is hard, we can at least get some idea why in practice we often observe that thick tails are desirable.

Furthermore, the theoretical result that the actual posterior distribution p(X) is the optimal importance function is derived based on an infinite number of samples. In practice, we can only afford a finite number of samples. In order that the samples effectively cover the whole support of the posterior distribution, we often need to make I(X) possess thicker tails than the posterior distribution. Suppose the mass of the tail area of the posterior distribution is ε and we draw a total of *N* samples. In order that the samples cover this area, we need at least one sample dropping in it, the probability of which is $p=1-(1-\varepsilon)^N$. In the case that $N\varepsilon \ll 1$, we have $p \sim N\varepsilon$. However, since N ε is very small, it is unlikely that any sample will drop in the tail area of p(X). Given the importance of covering the posterior probability distribution by I(X), we may deviate from the correct answer. For the probability to be greater than some value *u*, we have $N > u/\varepsilon$.

If we cannot afford the needed number of samples, we can instead increase the sampling density of I(X) in the tail area so that $\varepsilon > u/N$.

This is exactly why in practice importance functions with thicker tails than the actual posterior distribution often perform better [Geweke 1989]. Given that thick tails are desirable for importance sampling in Bayesian networks, we recommend the following strategy when designing an importance function.

First, we need to make sure that the support of I(X) includes that of p(X). Since Ω is compact and p(X) is finite for Bayesian networks, we only need to make sure that I(X)>0 whenever p(X)>0.

Second, we can make use of any estimation method to learn or compute I(X). Many importance sampling-based algorithms have been proposed for Bayesian networks. Based on the nature of the methods that the algorithms use to obtain the importance functions, we classify the algorithms into three families. The first family uses the prior distribution of a Bayesian network as the importance function, including the probabilistic logic sampling [Henrion 1988] and likelihood weighting [Fung 1989, Shachter1989] algorithms. The second family resorts to learning methods to learn I(X) including the self-importance sampling (SIS) [Shachter 1989], adaptive IS [Ortiz 2000], AIS-BN [Cheng & Druzdzel 2000], and dynamic IS [Moral 2003] algorithms. The third family directly computes I(X) in the light of both the prior distribution and the evidence, including the backward sampling [Fung 1994], IS [Hernandez 1998], annealed importance sampling [Neal 1998], and EPIS-BN algorithms [Yuan 2006]. Although much work has been done in this direction, potential for further work is still huge.

Note that the calculation of I(X) is essentially an approximate inference problem for Bayesian networks. There are many deterministic approximate inference algorithms that lack the guarantee to converge to the correct answers but can provide a satisfactory lower/upper bound efficiently, such as variational methods [Jordan 1998]. Nothing prevents us from using these methods to quickly get an estimation of the posterior distribution, which may be able to serve as a good I(X).

The last step, based on the discussion in the previous section, is to diagnose light tails and try to get rid of them to achieve thick tails. We review several existing heuristic methods for this purpose:

ε heuristic [Cheng 2000, Ortiz 2000] defines the tails of the joint probability distribution of a Bayesian network as the states with extremely small or extremely large probabilities. Therefore, it sets a threshold $\ ε $ and replaces any smaller probability in the conditional probability tables in the network by ε. At the same time, it compensates for this change by subtracting the difference from the largest probability in the same conditional probability distribution. The purpose is to spread the mass of the joint probability distribution in order to make it more flat. The other heuristic in AIS-BN – setting the probability distributions of the parents of evidence nodes to uniform – also has the similar effect.

IF-tempering [Yuan & Druzdzel 2004] instead of just adjusting I(X) locally, IF-tempering makes the original importance function I(X) more flat by tempering I(X). The final importance function becomes $I'(X) \sim I(X)^{1/T}$, where T (T>1) is the tempering temperature.

Rejection control [Liu 2001]: When I(X) is not ideal, importance sampling often produces random samples with very small weights. Rejection control adjusts I(X) by rejecting some samples and adjusting the weights of the accepted samples bringing I(X) closer to the target function p(X).

Pruned Enriched Rosenbluth Method [Grassberger 1997, Liang2002, Rosenbluth 1955] is also a sample population-based method, similar to rejection control. Rejection control is based on the observation that samples with extremely small weights do not play much role in the final estimation, but make the variance of sample weights large. However, there is yet another source of problem: samples with extremely large weights often overwhelmingly dominate the estimator and make other samples less effective. To eschew both problems, PERM assumes that the sample weights are built up in many steps and long range correlations between these steps are often weak and adjusts the samples accordingly.

Intentionally biased dynamic tuning [Cheng & Druzdzel 2000b, Ortiz 2000]. Dynamic tuning looks on the calculation of I(X) itself as a self-improving process. Starting from an initial I(X), dynamic tuning draws samples from the current I(X) and then use the samples to refine I(X) in order to obtain a new function. The new I(X) improves the old one at each stage. Dynamic tuning has been applied in several learning-based importance sampling algorithms. However, only two of them observe the importance of thick tails [Cheng & Druzdzel 2000, Ortiz 2000] and apply ε-cutoff to try to ensure that property in order to get better convergence rates.

Hybrid Loopy Belief Propagation algorithm

The Loopy Belief Propagation (LBP) algorithm, applied in our work on the EPIS-BN algorithm [Yuan & Druzdzel 2003], finds an approximation of the posterior marginal distributions for all nodes in a network by propagating local messages, and it has been extensively used as an approximate method for discrete models. However, the idea is general enough to be applicable to any networks. Sudderth et al. (2003) proposed the NBP algorithm extending belief propagation to deal with undirected continuous graphs.

As the next step in our work on stochastic sampling algorithms for general modeling systems, we extended the LBP and Nonparametric Belief Propagation (NBP) [Sudderth et al. 2003] algorithms to deal with general hybrid Bayesian networks. We call the resulting algorithm Hybrid Loopy Belief Propagation (HLBP). Our extension is general enough to deal with linear or nonlinear equations and arbitrary probability distributions and naturally accommodate the scenario where discrete variables have continuous parents. The main idea of HLBP is to represent each LBP message as a Mixture of Gaussians (MG) and formulate their calculation as Monte Carlo integration problems. The extension is far from trivial due to the enormous complexity brought by deterministic equations and mixture of discrete and continuous variables. Another advantage of the algorithm is that it can recover the true posterior distributions, unlike many existing approaches, which focus on the mean and standard variation only. We also propose an importance sampler to compute the product of MGs, whose accuracy is comparable to the Gibbs sampler in [Sudderth 2003]. Third, we propose a technique called lazy LBP to improve the efficiency of HLBP. Just as LBP, we anticipate that HLBP will work well for many practical models and can serve as a promising approximate method for hybrid Bayesian networks.

HEPIS-BN: Hybrid sampling algorithm for Bayesian networks

Because sampling methods are applicable to any probability distributions, they are excellent candidates for inference in general hybrid Bayesian networks, in which we have mixtures of discrete and continuous variables with arbitrary probability distributions or equations. However, not much work can be found in the literature on this topic except for algorithms for *conditional linear Gaussian* (CLG), a special instance of hybrid models. A possible reason is that it is difficult to come up with a good importance function for general hybrid models. To address these limitations, we proposed a new importance sampling algorithm called *Evidence Pre-propagated Importance Sampling algorithm for general hybrid Bayesian networks* (HEPIS-BN), whose main idea is to use *Hybrid Loopy Belief propagation* [Yuan & Druzdzel 2006] described in the previous section to calculate an importance function for importance sampling, following the idea originally proposed in [Yuan & Druzdzel 2003, Yuan & Druzdzel 2005]. The main advantage of the new algorithm is that it does not put any restrictions on the form of the hybrid Bayesian networks and is general enough to deal with hybrid Bayesian networks represented with arbitrary probability distributions or equations. Second, importance sampling-based algorithms provide the guarantee to converge to the correct posterior distributions given enough samples, unlike many existing approaches which only provide their first two moments. Only given the posterior distributions will we truly understand where most the mass locates. Third, our algorithm naturally accommodate the situation where discrete variables have continuous parents. We use generalized softmax to model this kind of relations. However, we can use other representations as well. For example, depending on values of the continuous parents, we can use different discrete distributions to model the discrete child. These kinds of modelling power enable us us to build more realistic models and aid decision making in more general settings.

Empirical results show that the HEPIS algorithm is a promising approach. We tested the algorithm on three small benchmark hybrid models and we typically observe that HEPIS-BN perform very well. More importantly, we observed that HEPIS-BN is very stable in face of unlikely evidence. This property makes HEPIS-BN a promising approach for addressing inference tasks in much larger real models.

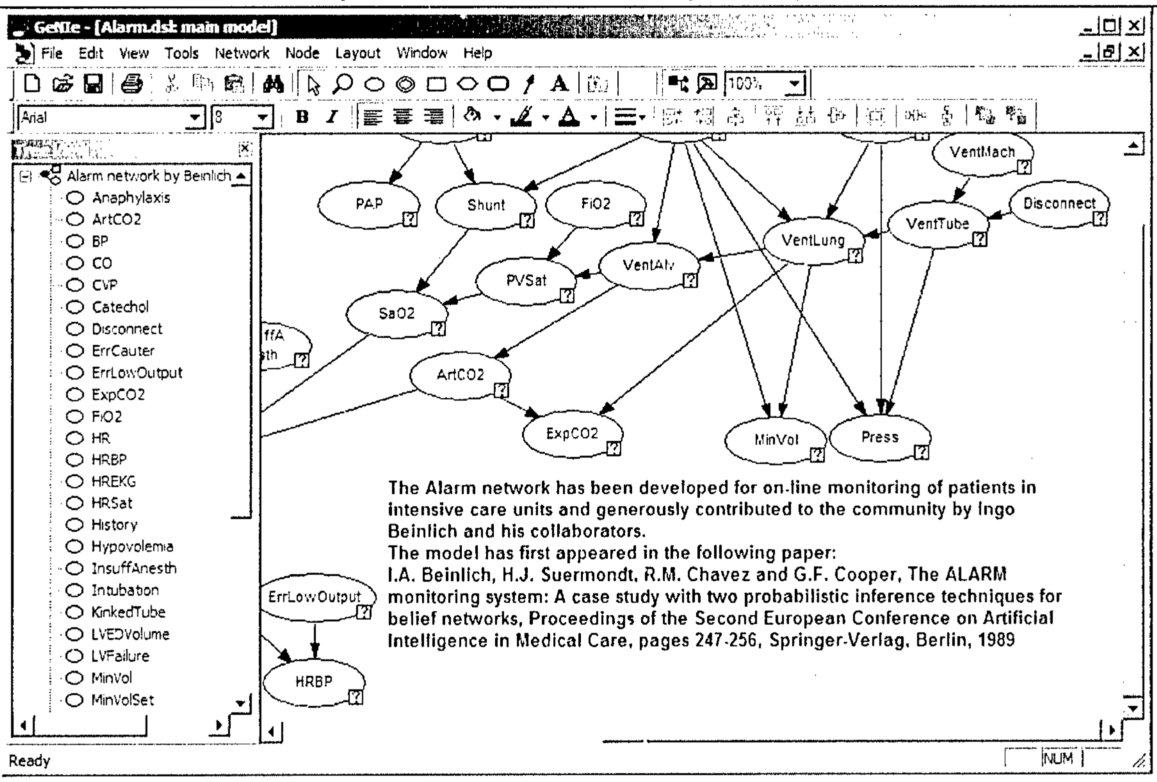
GeNIe and SMILE©

A major accomplishment of the project is the implementation of the system. Since there is much interest in Bayesian networks, influence diagrams, and decision-analytic systems, we have put much effort in making the implementation easy to use and robust and decided to share it with the community. We believe that this will bring a high payoff in the long run in terms of practical applications based on our system. This section summarizes the main elements of the software developed in the course of the project. More details can be found in our publications and in the comprehensive on-line manuals.

In the course of the project, 2003-2005 we have released the second generation of the program. We have made substantial progress in the speed of the algorithms (as of writing this final report, we have heard of the results of the competition organized by the 2006 Conference on Uncertainty in Artificial Intelligence, a prime conference in our area, in which **SMILE©** did extremely well, outperforming the other participating programs by typically between one and two orders of magnitude). We have updated our comprehensive on-line help for **GeNIe** (the user interface running on Windows machines), useful for both beginning modelers and students in decision-analytic methods and a documentation for **SMILE©** (**S**tructural **M**odeling, **I**nference, and **L**earning **E**ngine), a portable library of C++ classes for decision-theoretic reasoning, **GeNIe**'s reasoning engine. We have replaced **SmileX**, an Active-X control version of **SMILE©** that allows the program to be used from most Windows applications, including Visual Basic, Excel, and HTML pages, by **SMILE.NET**, which offers even more functionality. We have equipped all program components with comprehensive manuals. We made all our programs available on the World Wide Web in July 1998 (the address to download the program is: http://genie.sis.pitt.edu/). There are a growing number of users of our software. Over 10,000 people from countries all over the world downloaded it since the release date. We have heard very positive feedback from these users, especially those who have tried **GeNIe 2.0**.

Within the last year, we have developed a Java version of **SMILE©**, **jSMILE©**. We have also created a version of **SMILE©** for Pocket PC, which allows **GeNIe/SMILE©** applications to be run on palm-top computers.

Screenshots of **GeNIe 2.0** main model developer interface is presented below.

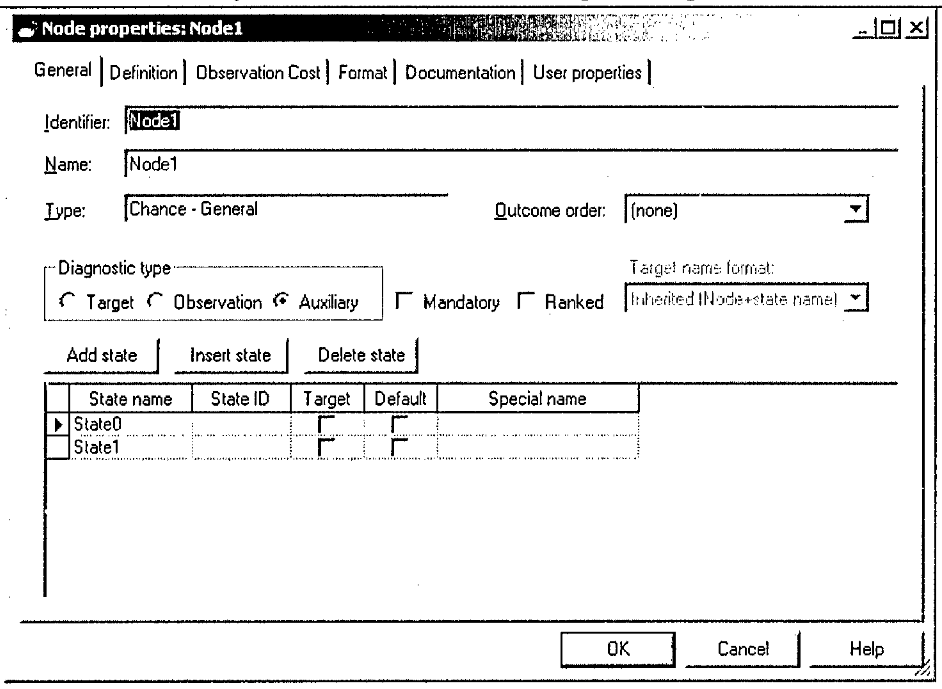


The Alarm network has been developed for on-line monitoring of patients in intensive care units and generously contributed to the community by Ingo Beinlich and his collaborators.
The model has first appeared in the following paper:
I.A. Beinlich, H.J. Suermondt, R.M. Chavez and G.F. Cooper, The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks, Proceedings of the Second European Conference on Artificial Intelligence in Medical Care, pages 247-256, Springer-Verlag, Berlin, 1989

Diagnostic software module

We have developed special features for diagnostic applications in **GeNIe** and **SMILE**©. One of these is computation of value of information for potential diagnostic tests and observations.

Each defect, error message, symptom, and test is represented as a separate node of a graphical causal model that is at the foundation of a Bayesian network model. Since a general purpose Bayesian network model does not make a distinction between the meanings of different types of nodes, the modeler has to indicate which of them represent defects, which are observations, and which are possible tests. For example, nodes representing components can have states labeled *Ok* and *Defective*. For nodes representing error messages, the states can be *Present* and *Absent*. For test nodes, the states could be labeled *Passed* and *Failed*. Nodes are connected together in the Bayesian network using directed links. The links typically follow the causal direction, i.e., go from the nodes that represent possible faults to nodes representing observations, error messages, tests, and symptoms. A link from a given component to a symptom can indicate that the symptom can be caused by the defects of the component. A link from a given component to a test can indicate that the test can be used to determine whether or not the component is defective. The following dialog box allows for entering diagnostic node properties.


Node properties: Node1
General | Definition | Observation Cost | Format | Documentation | User properties
Add state
Insert state
Delete state
OK
Cancel
Help

Identifier: Node1

Name: Node1

Type: Chance - General

Outcome order: (none)

Diagnostic type: ◯ Target ◯ Observation [✓ Auxiliary]

☐ Mandatory ☐ Ranked

Target name format: Inherited (Node+state name)

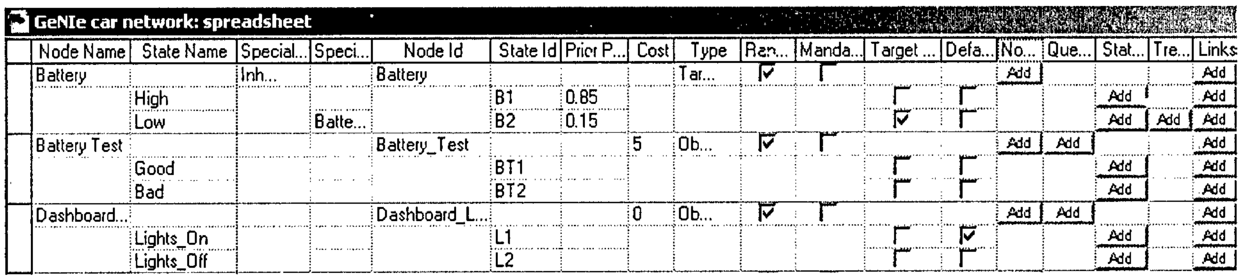
	State name	State ID	Target	Default	Special name
▶	State0		☐	☐	
	State1		☐	☐	

Spreadsheet View

The Spreadsheet window is a special extension of **GeNIe** that is useful in rapid model building – all properties of every variable of a model are listed in one window and the user specifying a model can move rapidly between variables and enter or modify their specifications.

The Spreadsheet window consists of several columns of information that describe the individual nodes (represented by rows) in detail. There are separate rows for each node and also for each state of the node. The information in the *Spreadsheet View* is also available in the *Node Properties Sheet*, however the *Node Properties Sheet* contains additional information that is not available in the Spreadsheet View, e.g., conditional probability tables.

The columns of the spreadsheet are described below in the order from left to right. You can directly jump to the description of a particular column by clicking on the column in the image below.



Node Name	State Name	Special...	Speci...	Node Id	State Id	Prior P...	Cost	Type	Ran...	Manda...	Target ...	Defa...	No...	Que...	Stat...	Tre...	Links
Battery		Inh...		Battery				Tar...	✓	☐			Add				Add
	High				B1	0.85					☐	☐			Add		Add
	Low		Batte...		B2	0.15					✓	☐			Add	Add	Add
Battery Test				Battery_Test			5	Ob...	✓	☐			Add	Add			Add
	Good				BT1						☐	☐			Add		Add
	Bad				BT2						☐	☐			Add		Add
Dashboard...				Dashboard_L...			0	Ob...	✓	☐			Add	Add			Add
	Lights_On				L1						☐	✓			Add		Add
	Lights_Off				L2						☐	☐			Add		Add

Treatment of Cost of Observation

GeNIe allows for entering the cost of observing the value of the current node. The cost of performing a test can be expressed on some scale, e.g., dollars or time in minutes. This scale must be used for all costs encoded in the model. The cost can be a combination of three different types of costs: simple costs, conditional costs, and group costs. Each of these different types of cost effect the test ranking in a special way.

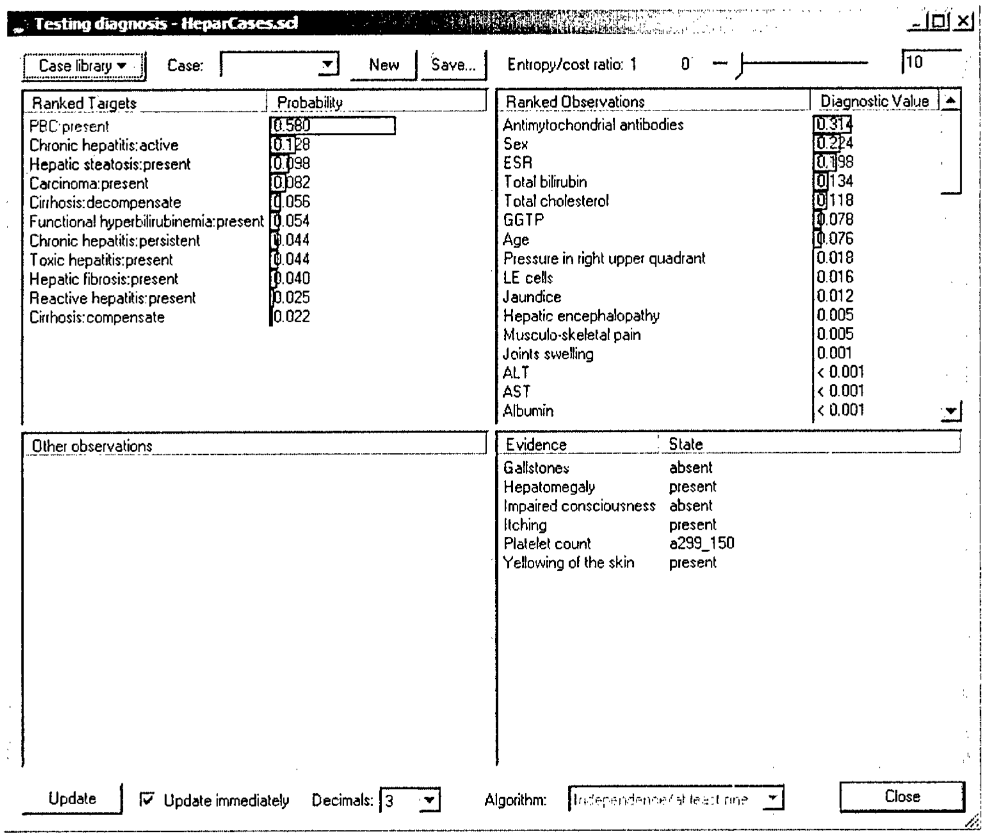
GeNIe uses two factors in evaluating the value of an observation. One is the observation's effectiveness in determining whether a given defect is likely. The other is the cost of performing the test. The cost of performing the test will affect the ranking by the following formula:

$$ValueOfObservation = CrossEntropy(test) - alpha * cost$$

where *alpha*, also known as "entropy to cost ratio," is a user-defined constant that brings the measures of informativeness to a common scale. The formula essentially weights costs and a measure of informativeness (*CrossEntropy*) linearly with a weighting factor *alpha.*

After computing the *ValueOfObservation* for each unobserved test in the model, **GeNIe** sorts the test according to the decreasing value. Zero cost associated with a test means that there is no cost associated with observation. The effect will be the the test will be ranked only by a measure of informativeness. Simple and conditional costs are for *Ranked* and *Observable* nodes. Group costs, on the other hand, are for *Ranked* nodes set at *Unobservable* and *Auxiliary* nodes.

The value of *alpha* is set in the diagnostic window by moving the slider in the upper-right hand side of the window. The upper bound of *alpha* can be modified by typing the new value in the small edit box to the right of the slider.



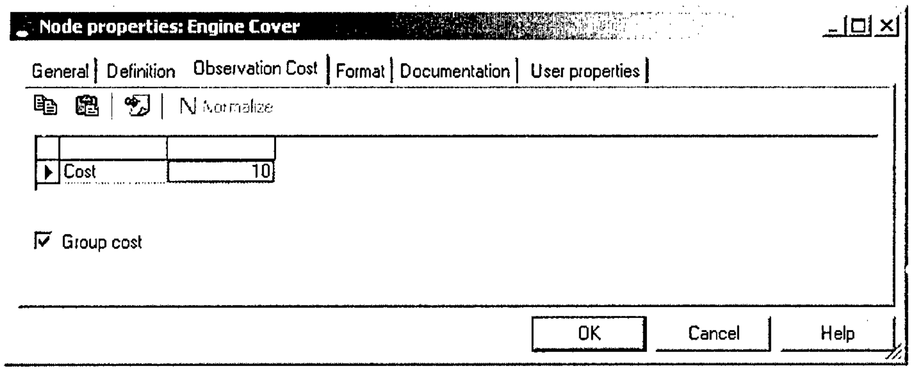
Ranked Targets	Probability
PBC:present	0.580
Chronic hepatitis:active	0.128
Hepatic steatosis:present	0.098
Carcinoma:present	0.082
Cirrhosis:decompensate	0.056
Functional hyperbilirubinemia:present	0.054
Chronic hepatitis:persistent	0.044
Toxic hepatitis:present	0.044
Hepatic fibrosis:present	0.040
Reactive hepatitis:present	0.025
Cirrhosis:compensate	0.022

Ranked Observations	Diagnostic Value
Antimytochondrial antibodies	0.314
Sex	0.224
ESR	0.198
Total bilirubin	0.134
Total cholesterol	0.118
GGTP	0.078
Age	0.076
Pressure in right upper quadrant	0.018
LE cells	0.016
Jaundice	0.012
Hepatic encephalopathy	0.005
Musculo-skeletal pain	0.005
Joints swelling	0.001
ALT	< 0.001
AST	< 0.001
Albumin	< 0.001

Evidence	State
Gallstones	absent
Hepatomegaly	present
Impaired consciousness	absent
Itching	present
Platelet count	a299_150
Yellowing of the skin	present

Simple costs

Simple cost is used when the cost of observing a node is independent of whether other nodes are observed or not. Simple cost is also known as the cost associated with actually performing the test. For example, a simple cost could be the cost of performing a simple test. This cost is set in the Node Properties sheet under the Observation cost tab. Simple costs can also be entered in the Spreadsheet window.



Cost	10

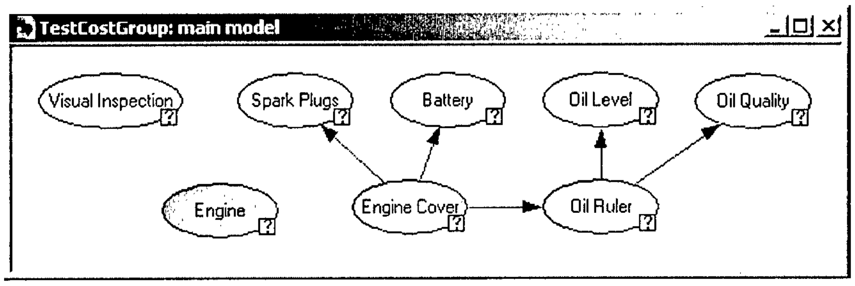
☑ Group cost

GeNIe models that include observation costs define two overlapping networks. The first is a regular Bayesian network. This network represents the structure of the problem along with uncertainties, encoded by means of probabilities. The second network is the cost network, can be easily recognized by the fact that all arcs in it are colored red, This network represents the conditional cost structure.

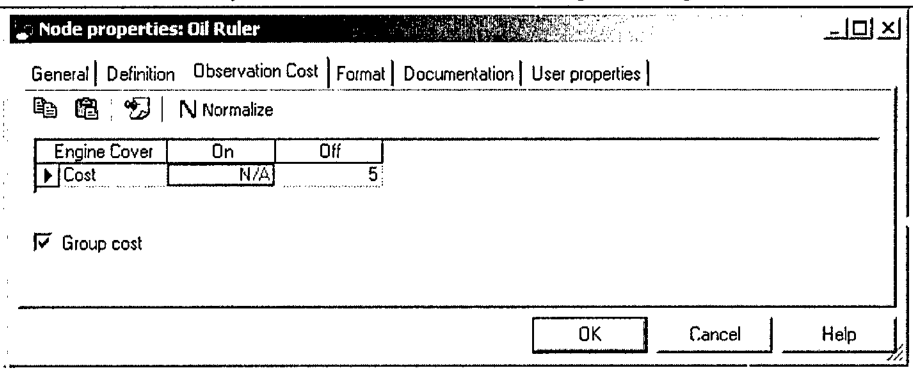
Conditional costs

Conditional cost is the cost associated with performing the test where the actual cost for a test node will depend on the state of its parent nodes. For example, if a component is broken and for it to be fixed another component has to be removed, the cost will then be dependent upon both of the components. The cost of measuring some parameter of a locomotive engine depends on whether the locomotive is in the shop or in the field. It may be much lower when the locomotive is in the shop.

Shown below is a cost network for the inspection of the engine of a car:



The node Engine has normal arcs connecting it to the other nodes, but since this is a Cost Graph View they are not displayed. The cost of inspection of the nodes Oil Ruler, Spark Plugs and Battery is dependent upon the status of the Engine Cover. For example, for the Oil Ruler, if the Engine cover is on, then the inspection cannot be done, if the engine cover is off, then the cost of using the Oil Ruler is 5.


Node properties: Oil Ruler
General | Definition | Observation Cost | Format | Documentation | User properties
Normalize
OK
Cancel
Help

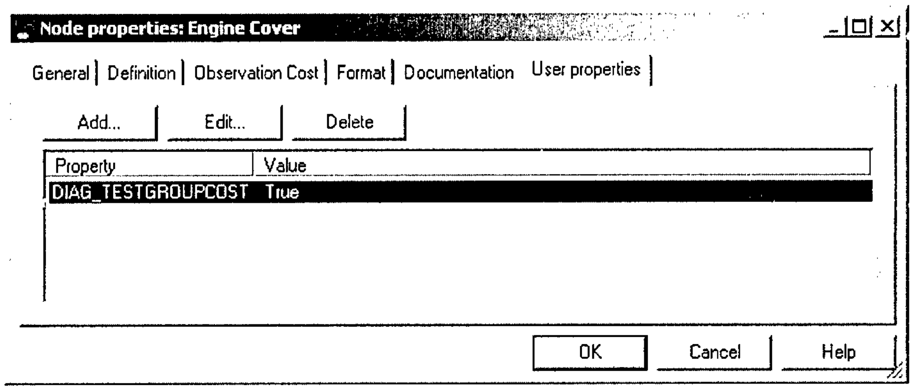
Engine Cover	On	Off
Cost	N/A	5

☑ Group cost

Group costs

Group cost is used when several tests are performed in a group. A typical example of a group cost is taking a blood sample (the cost of the blood sedimentation rate test incurs the cost of first drawing a blood sample from a patient) or performing a test of an internal part of a locomotive engine (which involves removing the engine cover of a locomotive to access engine parts). Once a blood sample is drawn or the locomotive cover is open, any other tests will incur only simple costs. A group of nodes, with a common group cost can be defined in the *Node properties* window. *Group cost* is one time cost associated with performing the first test of a particular group. The group cost node also designates the cost group.

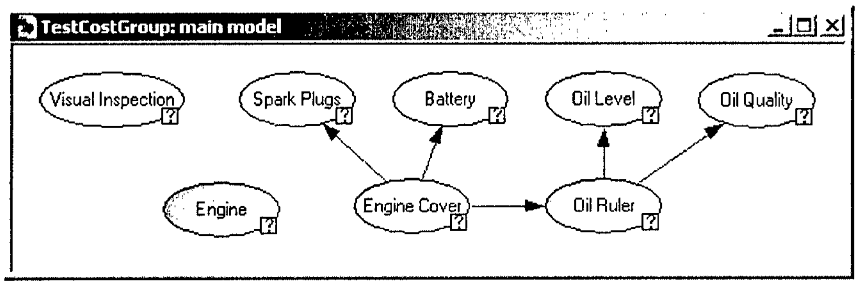
To add a group cost for a node, the user must go to the User Properties tab and *Add* a property named DIAG_TESTGROUPCOST. This is shown below:


Node properties: Engine Cover
General | Definition | Observation Cost | Format | Documentation | User properties
Add...
Edit...
Delete
OK
Cancel
Help

Property	Value
DIAG_TESTGROUPCOST	True

Group costs are entered in a fashion similar to simple costs and conditional costs.

In the figure below, the node *Engine Cover* is a node that distinguishes the group that is being observed.



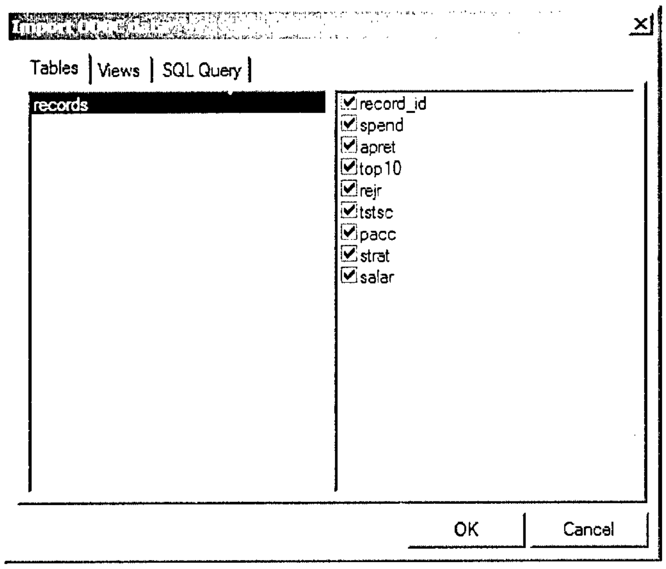
The nodes that are the children of *Engine Cover* will incur the cost associated with opening the *Engine Cover*. As mentioned, this cost will only be applied to child nodes as long as none of the nodes of that group have been instantiated. Assume that the group cost of opening the *Engine Cover* is set to 10 units. In the example, the cost of observing the state of *Spark Plugs* is set at 2 units and the cost of checking the *Battery* is set at 5 units. As long as neither of the nodes is observed, the total cost of *Spark Plugs* is 12 units and the cost of *Battery* is 15 units. If either node, *Spark Plugs* or *Battery*, is set to a state, then the group cost will be set to 0 units. For example, if *Battery* is set to pass, the cost of performing the *Spark Plugs* test will now be 2 units. The 10 unit cost associated with the node *Engine Cover* no longer applies.

Learning and data mining software module

The learning and data mining module of **GeNIe** and **SMILE**© allow for learning probabilistic models and their elements from data. This functionality becomes increasingly important as measurements are being performed and stored, as cases become accumulated, and as models are being built for domains that are data intensive. The module allows for accessing data from both data bases and text files, preparing the data for learning or data mining, and, finally for learning model structure and parameters. The module is useful not only for those users who apply probabilistic methods but also to those users who apply other data mining techniques.

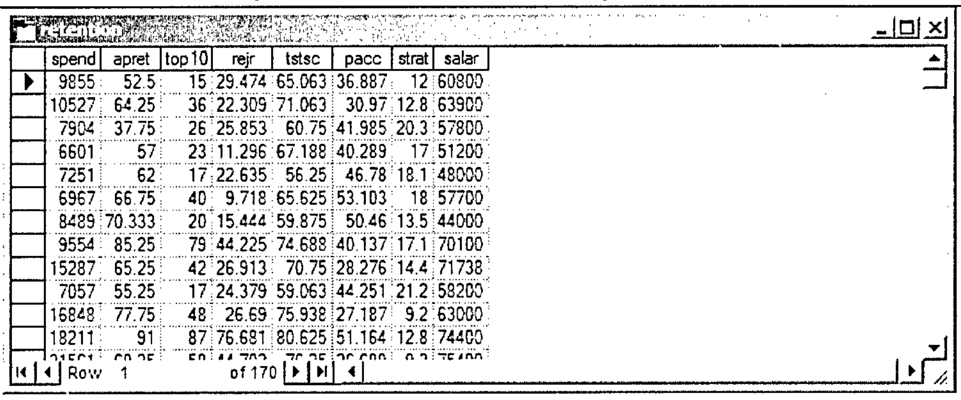
Accessing and pre-processing data

The learning module allows reading data from text files (collections of records with comma, space, or tab-delimited fields, also compatible with Microsoft Excel text format) and retrieving them from ODBC data bases. The latter includes simple SQL queries, composing virtual records from existing data base tables, etc. Below is a simple **GeNIe** interface for querying a database:



✓ record_id
✓ spend
✓ apret
✓ top10
✓ rejr
✓ tstsc
✓ pacc
✓ strat
✓ salar

GeNIe uses the data grid view to display the loaded data files and let's users work with them much like with spreadsheets. The figure below shows a data grid:

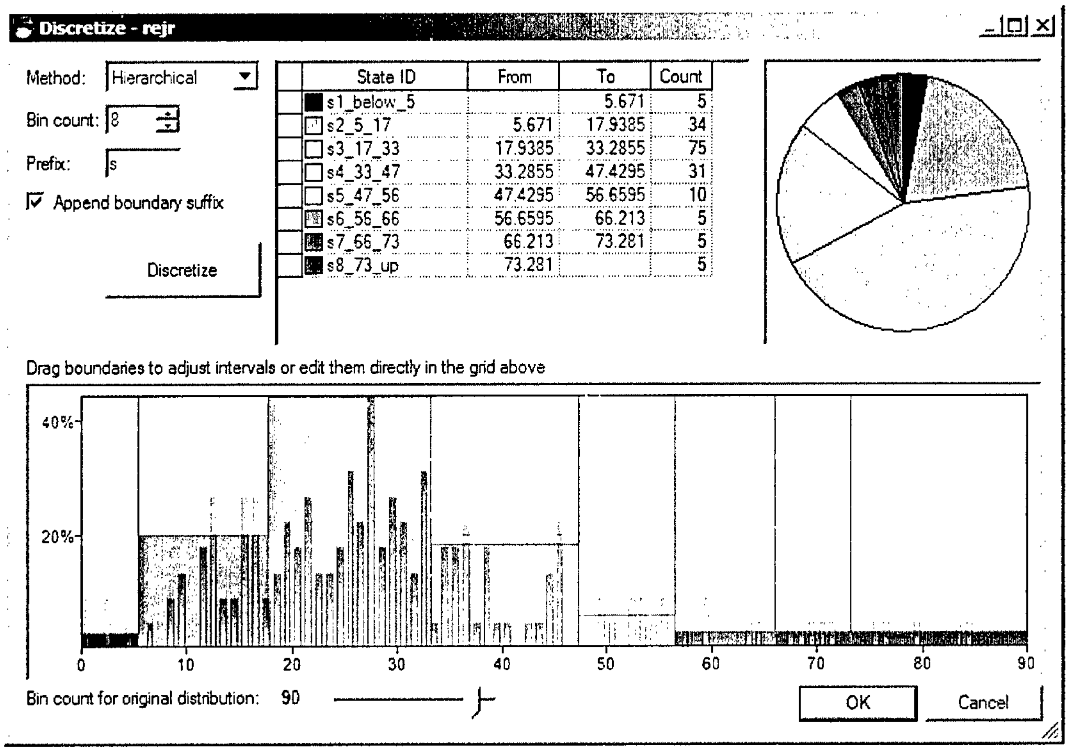


	spend	apret	top10	rejr	tstsc	pacc	strat	salar
▶	9855	52.5	15	29.474	65.063	36.887	12	60800
	10527	64.25	36	22.309	71.063	30.97	12.8	63900
	7904	37.75	26	25.853	60.75	41.985	20.3	57800
	6601	57	23	11.296	67.188	40.289	17	51200
	7251	62	17	22.635	56.25	46.78	18.1	48000
	6967	66.75	40	9.718	65.625	53.103	18	57700
	8489	70.333	20	15.444	59.875	50.46	13.5	44000
	9554	85.25	79	44.225	74.688	40.137	17.1	70100
	15287	65.25	42	26.913	70.75	28.276	14.4	71738
	7057	55.25	17	24.379	59.063	44.251	21.2	58200
	16848	77.75	48	26.69	75.938	27.187	9.2	63000
	18211	91	87	76.681	80.625	51.164	12.8	74400
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Row 1 of 170

The columns denote the variables of the underlying model and rows contain combinations of their states.

Before structure (or only parameters) is learned, it might be necessary or desirable to prepare the data and to perform operations like identifying and removing outliers, discretization, dealing with missing data, verifying distributional assumptions, etc. The software allows for selecting missing values or all records that contain missing values, deleting these records, or choosing an algorithm that replaces the missing values. **GeNIe** offers a tool for discretize values. Several automatic methods of discretization are provided, such are hierarchical, uniform widths, or uniform counts. The user can select the number of bins for the original distribution using the slider at the bottom of the window. Here is a screen shot of the discretization interface:



Discretize - rejr

Method: Hierarchical

Bin count: 8

Prefix: s

[✓] Append boundary suffix

	State ID	From	To	Count
	■ s1_below_5		5.671	5
	□ s2_5_17	5.671	17.9385	34
	□ s3_17_33	17.9385	33.2855	75
	□ s4_33_47	33.2855	47.4295	31
	□ s5_47_56	47.4295	56.6595	10
	▨ s6_56_66	56.6595	66.213	5
	▨ s7_66_73	66.213	73.281	5
	■ s8_73_up	73.281		5

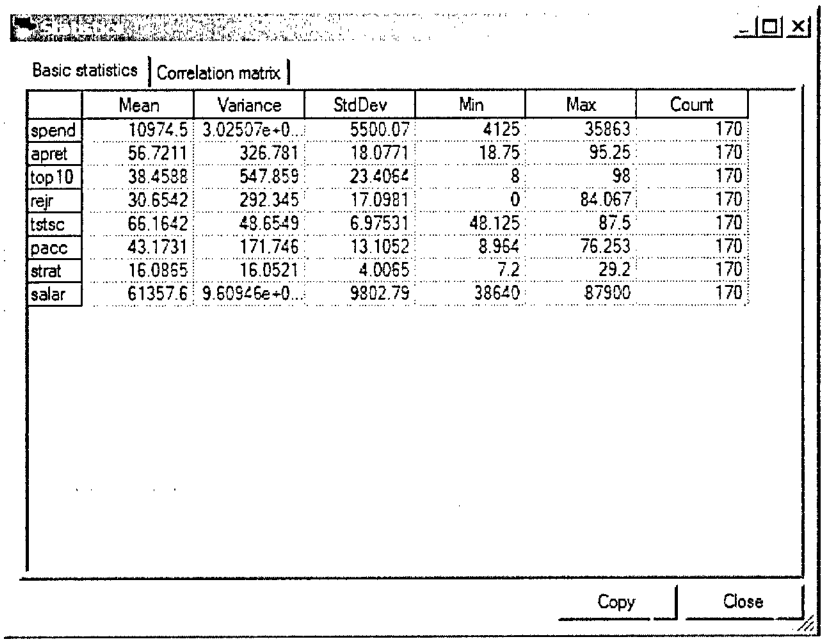
Drag boundaries to adjust intervals or edit them directly in the grid above

Bin count for original distribution: 90

A unique feature of this interface is that the discretization can be tuned up manually by simply dragging the boundaries with the mouse.

Another useful feature of the data preparation interface is the possibility of merging states. This will typically be used when the data file contains several labels for the same outcome, such as M, Man, and Male for a patient's sex.

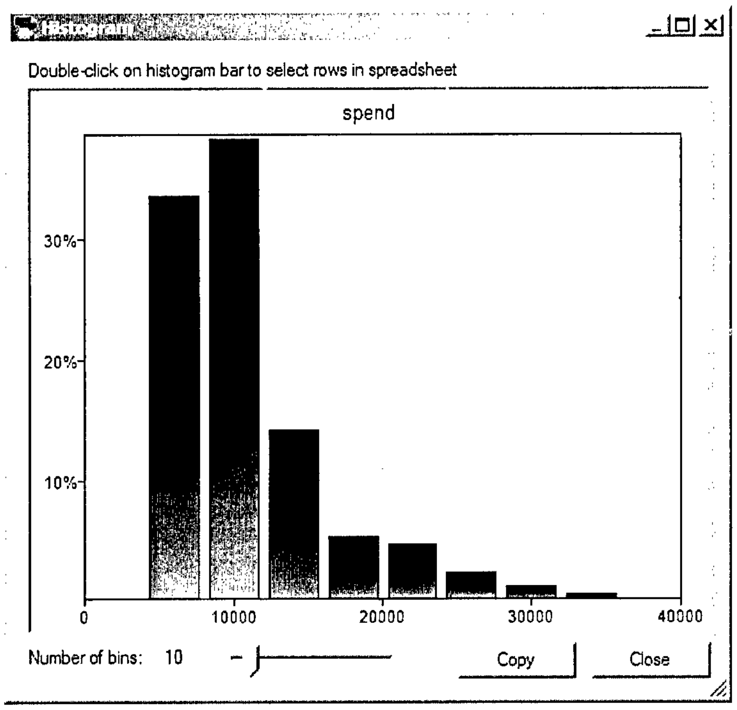
The program allows exploration of the data by computing and displaying simple statistics, including moments, min and max values, etc.



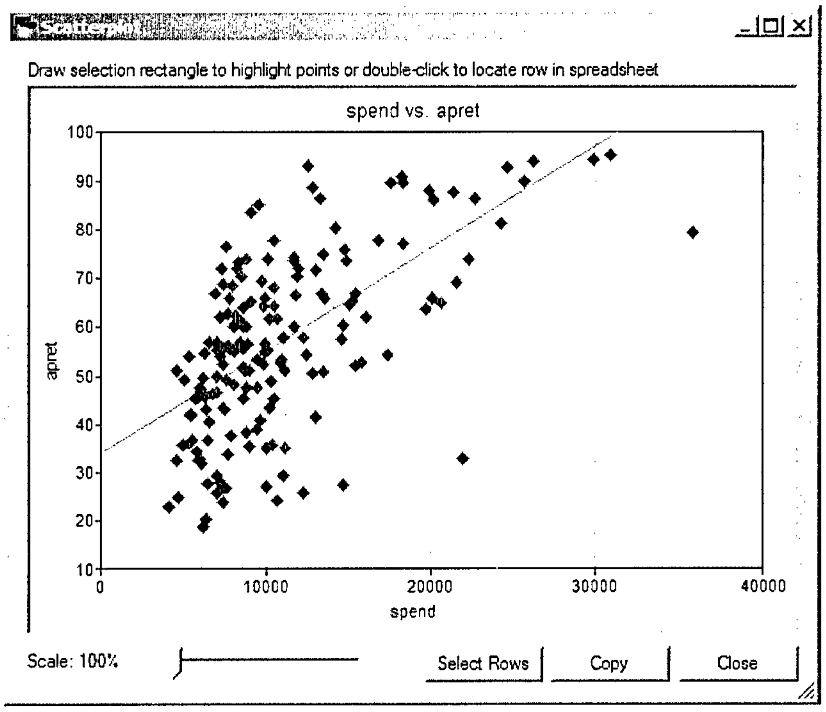
	Mean	Variance	StdDev	Min	Max	Count
spend	10974.5	3.02507e+0...	5500.07	4125	35863	170
apret	56.7211	326.781	18.0771	18.75	95.25	170
top10	38.4588	547.859	23.4064	8	98	170
rejr	30.6542	292.345	17.0981	0	84.067	170
tstsc	66.1642	48.6549	6.97531	48.125	87.5	170
pacc	43.1731	171.746	13.1052	8.964	76.253	170
strat	16.0855	16.0521	4.0065	7.2	29.2	170
salar	61357.6	9.60946e+0...	9802.79	38640	87900	170

The second tab in this dialog allows for examining the correlation matrix among the various variables in the data set. This is useful in case of continuous data and allows to see a measure of linear dependence among the variables. Many learning algorithms assume multi-normality of the data.

To see the distribution of the values of a variable (a column in the data set), users can invoke the histogram feature. Our histogram is unique in that it allows to change the bin size interactively. The shape of a histogram is known to depend strongly on the bin size but to our knowledge there is no software that allows changing this size interactively. **GeNIe**'s histogram interface is shown below:



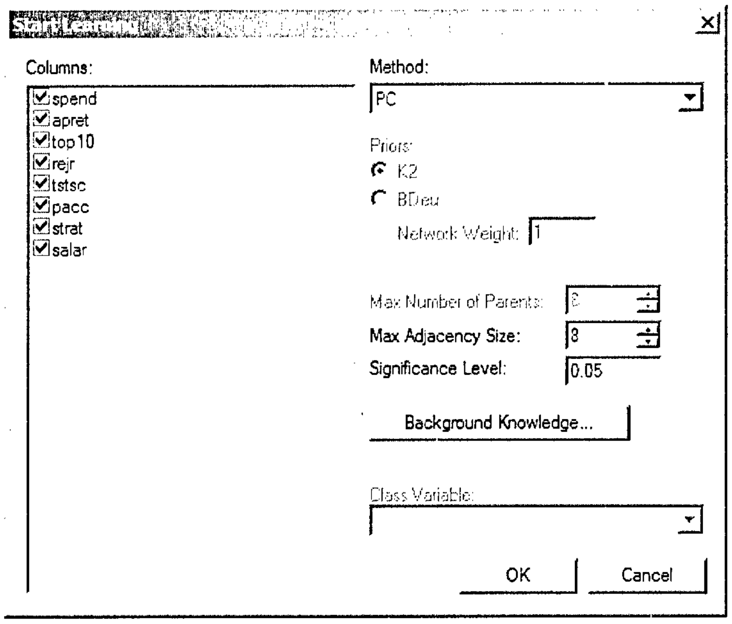
Another extremely useful feature in exploratory analysis of data is a scatter plot. It allows a user to see how precisely two variables co-depend. **GeNIe**'s scatter plot interface is shown below:


Scatterplot
Draw selection rectangle to highlight points or double-click to locate row in spreadsheet
spend vs. apret
apret
spend
Scale: 100%
Select Rows
Copy
Close

Double click a point to select the corresponding data row in the data grid.

Learning the model structure

The learning module allows for learning the model structure from data. The user can select variables for learning, an algorithm to be applied, and enter outside knowledge, useful in determining causal influences and orienting causal arcs. The basic structure learning interface is shown below.



Columns:
☑ spend
☑ apret
☑ top10
☑ rejr
☑ tstsc
☑ pacc
☑ strat
☑ salar

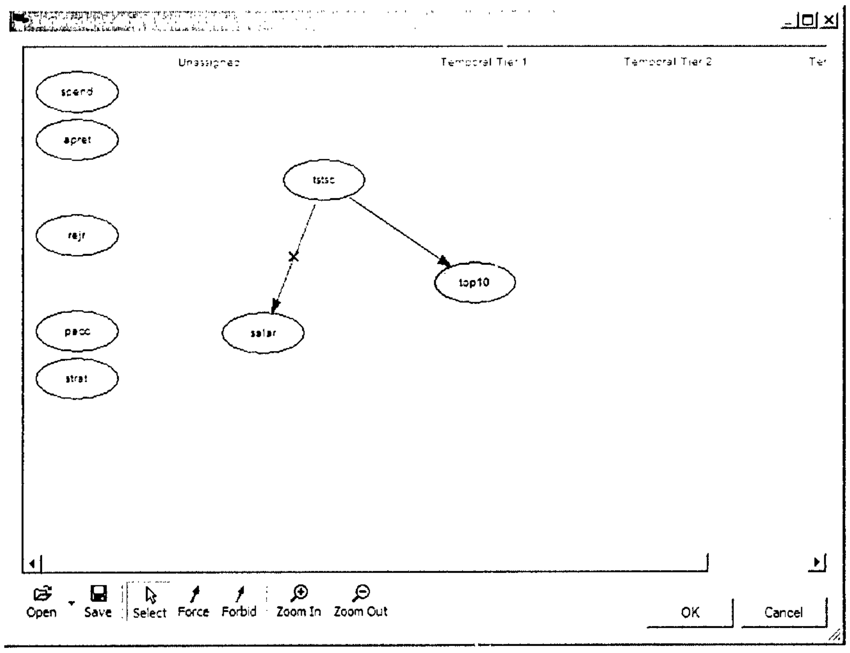
Method: PC

Priors:
[✓ K2]
BDeu
Network Weight: 1

Max Number of Parents: 8
Max Adjacency Size: 8
Significance Level: 0.05

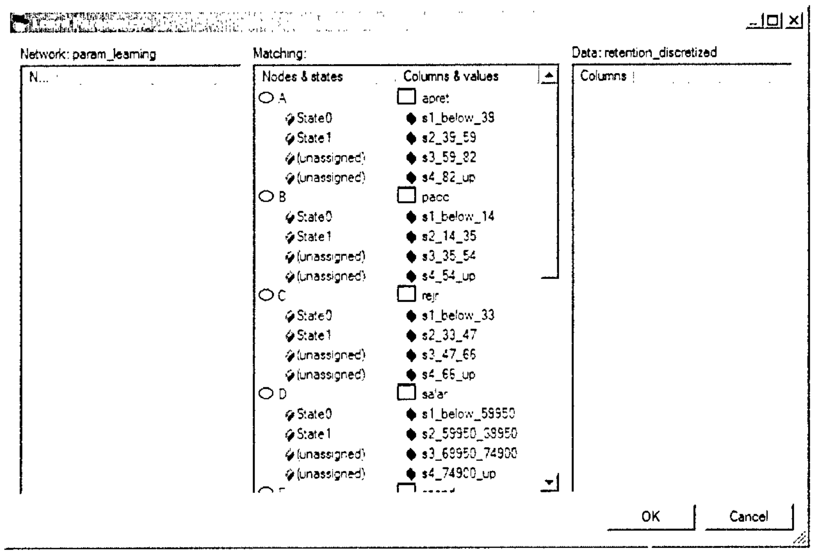
Class Variable:

Background Knowledge Editor, which can be invoked from the main structure learning window allows for forcing an arc, forbidding an arc, and assign variables to temporal tiers, which helps in orienting causal arcs (the assumption here is that causality does not work backward in time). Background knowledge can be saved and retrieved later, which is useful in case of learning from different data files or from different versions of the same data file.



Learning the model structure

GeNIe allows for learning parameters of an existing network (i.e., with the structure already defined). A simple interface, pictured below, allows for creating a mapping between the variables defined in the network (left column) and variables defined in the data set (right column).


Network: param_learning

Matching:

Data: retention_discretized

Nodes & states	Columns & values
A	apret
State0	s1_below_39
State1	s2_39_59
(unassigned)	s3_59_82
(unassigned)	s4_82_up
B	pacc
State0	s1_below_14
State1	s2_14_35
(unassigned)	s3_35_54
(unassigned)	s4_54_up
C	rejr
State0	s1_below_33
State1	s2_33_47
(unassigned)	s3_47_66
(unassigned)	s4_66_up
D	salar
State0	s1_below_59950
State1	s2_59950_69950
(unassigned)	s3_69950_74900
(unassigned)	s4_74900_up

To create a mapping, the user simply drags and drops variables or their states (this is useful in case the states in the network have different names than the states in the data file). All the established mappings are shown in the middle column. If the names of variables and states are the same or similar, **GeNIe** initializes the dialog with pre-matched mappings, which can be subsequently modified by the user.

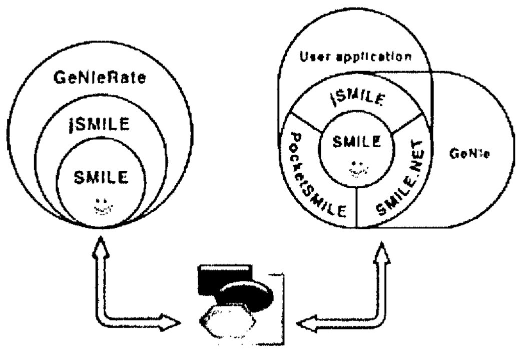
Other contributions

GeNIeRate: an interactive generator for very large diagnostic models

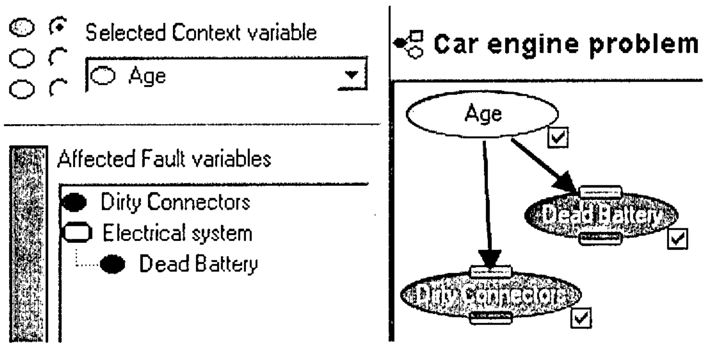
While the existing diagnostic BN models perform very well, the technique is still not widely used and accepted. One of the main reasons for this is that building a BN model is a laborious and time consuming task. During the model building process, both the qualitative and quantitative parts of the BN have to be designed. This can be done in three different ways: (1) both structure and parameters can be learned from data without human interaction, (2) consulting a domain expert to design the structure and the parameters, or (3) combine learning from data and consulting a domain expert. In order to learn successful diagnostic BN models from data one would need a very large data set, which is almost never available for diagnostic models. So building a diagnostic BN model will most of the time come down to an interaction of a knowledge engineer and a domain expert. They will consult technical manuals, test procedures, and repair databases to define the variables in that domain, determine the interactions among them, and to elicit the parameters. To give an idea of the time used to design a diagnostic BN model: the construction of the HEPAR-II model [Onisko, Druzdzel & Wasyluk 2001] used to diagnose liver disorders took approximately 300+ hours of which roughly 50 hours were spent with domain experts. The model consisted of 73 variables and its numerical parameters were learned from a data set of patient cases. One of the first large diagnostic systems that used Bayesian networks was the QMR-DT system [Shwe *et al.*, 1991]. This system was a probabilistic reformulation of the Quick Medical Reference (QMR), a rule-based system based on the INTERNIST-1 knowledge base developed at the University of Pittsburgh [Miller *et al.*, 1982]. The QMR-DT system contains approximately 5000 variables. The authors made several simplifying assumptions to deal with the complexity of constructing this network. The system was a BN with only two levels of variables representing two different types of variables: *diseases* and *findings*. The structure was such that the *disease* variables influence the outcome of the *findings*. While this structure was simple, its performance was close to the original QMR. But since the users of the system knew the independence assumptions the inconsistencies of QMR-DT could easily be explained.

Inspired by the QMR-DT system, we proposed a methodology to make diagnostic BN models in an intuitive and fast way for users who are not necessarily experts in Bayesian networks. Using our methodology, a domain expert will be able to build a model without any interaction with a knowledge engineer. Skaanning [2000] addresses the same problem applied in an application called *BATS Author*. The BATS Author hides the causality of the models from the user and also has the assumption that only one single fault can occur. Our methodology supports multiple faults and shows the causal structure of the graph. We believe that this will support the user in understanding what she is doing while building a diagnostic BN model. To keep the model building process simple for the user, the methodology is based on some simplifying assumptions. The first assumption is in the qualitative part of the model. The structure is such that it can only consist of three levels of variables. The top level represents *context* variables, the middle level represents the *fault* variables and the bottom level represents *evidence* which are the effects of when a *fault* occurs. Relations are only allowed in a top-down way: from *contexts* to *faults* and from *faults* to *evidences*. The second simplification is in the quantitative part. All the variables are modeled using Noisy-MAX gates. A Noisy-MAX gate is a generalization of the Noisy-OR gate for multi-valued variables. The goal of using the Noisy-MAX gate is to reduce the size of the CPT's of the variables. It implies that the number of parameters that have to be elicited by the user decreases from exponential to linear in the number of parents of that variable. Research shows that many interactions in practical models can be approximated by the Noisy-MAX gates [Zagorecki and Druzdzel 2004]. With these simplifications we sacrifice some theoretical modeling power and precision but we believe, and we plan to test this in practice, that the resulting models will not be significantly less accurate while being much easier to build.

Our proposed methodology is implemented in an application named: **GeNIeRate**. **GeNIeRate** is built on the top of jSMILE, a version of SMILE with a Java wrapper. The architecture of the program is shown below:



A fragment of the user interface of genierate is shown bwlow.



Proof of Correctness of the Causal Ordering Algorithm

We examined in detail the algorithm of Simon [1953], called the *causal ordering algorithm* (COA), used for constructing the causal ordering of a system given a complete specification of the system in terms of a set of structural equations that govern the variables in the system. This algorithm constructs a graphical characterization of the model in the form of a *directed cluster graph.* Simon argued in [Simon, 1953] and subsequent papers that a graph so generated explicates causal structure among variables in the model. We further formalize this claim and prove a theorem which justifies it. In particular, we prove that given a set of equations *E* satisfying certain conditions, any directed causal graph that is imposed on the variables consistent with *E* must be consistent with the graph generated by COA. Our paper describing this work has been accepted for publication by the *Artificial Intelligence Journal* (Dash & Druzdzel, 2006).

Annealed MAP

Maximum a Posteriori assignment (MAP) is the problem of finding the most probable instantiation of a set of variables given the partial evidence on the other variables in a Bayesian network. MAP has been shown to be a

NP-hard problem [Shimony 1994], even for constrained tree width networks, such as polytrees [Park 2002]. Exact approaches often fail to yield any results for MAP problems in extremely large Bayesian networks, and even approximate approaches may not yield efficient solutions. Previous approaches often fail to yield any results for MAP problems in large complex Bayesian networks. To address this problem, we proposed the Annealed MAP algorithm, a simulated annealing-based MAP algorithm (Yuan, Lu & Druzdzel, 2004). The Annealed MAP algorithm simulates a non-homogeneous Markov chain whose invariant function is a probability density that concentrates itself on the modes of the target density. We tested this algorithm on several real Bayesian networks. The results show that, while maintaining good quality of the MAP solutions, the AnnealedMAP algorithm is also able to solve many problems that are beyond the reach of previous approaches.

We have also introduced the Dynamic Weighting A* (DWA*) search algorithm for solving MAP [Sun, Yuan & Druzdzel, 2006]. By exploiting asymmetries in the distribution of MAP variables, the algorithm is able to greatly reduce the search space and yield MAP solutions with high quality. Typically, a small fraction of all possible assignments can be expected to cover a large portion of the total probability space with the remaining assignments having practically negligible probability [Druzdzel 1994]. Also, the DWA* uses dynamic weighting based on greedy guess [Park 2001, Yuan 2004] as the heuristic function. While it is theoretically not admissible (admissible heuristic should offer an upper bound on the MAP), it offers ε-admissibility and excellent performance in practice.

A Robust Independence Test for Constraint-Based Learning of Causal Structure

We proposed a method that combines ideas from Bayesian learning, Bayesian network inference, and classical hypothesis testing to produce a more reliable and robust test of independence for constraint-based (CB) learning of causal structure [Dash & Druzdzel 2003]. Our method produced a smoothed contingency table NXY Z that can be used with any test of independence that relies on contingency table statistics. NXY Z can be calculated in the same asymptotic time and space required to calculate a standard contingency table, allows the specification of a prior distribution over parameters, and can be calculated when the database is incomplete. We provided theoretical justification for the procedure, and with synthetic data we demonstrated its benefits empirically over both a CB algorithm using the standard contingency table, and over a greedy Bayesian algorithm. We showed that, even when used with non-informative priors, it results in better recovery of structural features and it produces networks with smaller KL-Divergence, especially as the number of nodes increases or the number of records decreases. Another benefit is the dramatic reduction in the probability that a CB algorithm will stall during the search, providing a remedy for an annoying problem plaguing CB learning when the database is small.

Sensitivity of Bayesian networks to imprecision in numerical parameters

We have conducted an empirical study that focused on the problem of sensitivity of Bayesian networks to imprecision in their parameters (Onisko & Druzdzel, 2003).

While most knowledge engineers believe that the quality of results obtained from Bayesian networks is not too sensitive to imprecision in probabilities, this remains a conjecture with only modest empirical support. Our earlier work on a Bayesian network model for diagnosis of liver disorders, Hepar II [Onisko, Druzdzel &

Wasyluk 2001], presented us with an excellent opportunity to shed some light on this question in a practical setting. We conducted an empirical study in which we systematically introduced noise in Hepar II's probabilities and tested the diagnostic accuracy of the resulting model. Similarly to Pradhan et al. [1996], we assumed that the original set of parameters and the model's performance are ideal. Noise in the original parameters leads to deterioration in performance. The main result of our analysis is that noise in numerical parameters starts taking its toll from the very beginning and not, as suggested by Pradhan et al., only when it is very large. Because Hepar II is a medical diagnostic model, we also study the influence of noise in each of the three major classes of variables: (1) medical history, (2) physical examination, (3) laboratory tests, and (4) diseases, on the diagnostic performance. Although the differences here were rather small, it seemed that noise in the results of laboratory tests was most influential for the diagnostic performance of our model. While our result is merely a single data point that sheds light on the hypothesis in question, our result suggested that Bayesian networks may be more sensitive to the quality of their numerical parameters than popularly believed.

An Efficient Sampling Algorithm for Influence Diagrams

We have also developed an efficient stochastic sampling-based algorithm for solving influence diagrams (Garcia-Sanchez & Druzdzel, 2004). We showed how sampling algorithms can be utilized very efficiently within an indirect evaluation method. Evaluating all decision strategies on the same set of random samples allowed for saving a significant amount of computation and also produced high quality anytime behavior. The algorithm we proposed is suitable for very large decision models for which exact algorithms are not feasible. Recent advances in stochastic sampling algorithms for BNs (e.g., (Cheng and Druzdzel, 2000; Yuan and Druzdzel, 2003)) have made it possible to perform inference in very large models with unlikely evidence. Because our algorithm can be based on any simulation algorithm for BNs, it allows for taking advantage of these developments. The algorithm is general in the sense of admitting full IDs with multiple decision and value nodes and containing observations. As a side-product, the algorithm computes also the posterior probability distribution of every chance node conditional on each of the possible strategies. For example, in a system that helps to decide between medication and surgery, the algorithm computes not only the expected utility of both choices, but also the probability of the patient dying in each case. We have found that this information is useful in an interactive modeling environment. The algorithm is, to our knowledge, the only algorithm developed for ID that exhibits anytime behavior when the ID has an arbitrary number of decision and utility nodes. Finally, the algorithm is very efficient and this is achieved by reusing random samples. Practically, only one random number per node is generated for each sample of the network. This random number is used to evaluate the probability distribution of the node conditional on each of the strategies.

Publications acknowledging support from AFOSR

Under review:

Adam Zagorecki and Marek J. Druzdzel (draft). Noisy Average Model for Local Probability Distributions

Changhe Yuan and Marek J. Druzdzel (2006). How Heavy Should Heavy Tails Be? Submitted to *International Journal of Approximate Reasoning.*

F. Javier Diez and Marek J. Druzdzel (2006). Canonical probabilistic interaction models. Submitted to *Knowledge Engineering.*

Adam Zagorecki and Marek J. Druzdzel (2006). How Common are Noisy-MAX Distributions in Practice?, Submitted to *IEEE Transactions on Data and Knowledge Engineering*

Journals:

Changhe Yuan and Marek J. Druzdzel (2004). Importance Sampling Algorithms for Bayesian Networks: Principles and Performance. *Mathematical and Computer Modelling*, 43(9-10):1189-1207, May 2006.

Marek J. Druzdzel and F. Javier Diez (2003). Combining Knowledge from Different Sources in Probabilistic Models. *Journal of Machine Learning Research*, 4(July):295-316, 2003.

Denver H. Dash and Marek J. Druzdzel. The correctness of the causal ordering algorithm. To appear in *Artificial Intelligence Journal.*

Major peer reviewed conferences:

Adam Zagorecki and Marek J. Druzdzel. Knowledge engineering for building Bayesian networks: How common are Noisy-MAX distributions in practice? To appear in the *Proceedings of the 17th European Conference on Artificial Intelligence*, Riva del Garda, Italy, August 28th - September 1st, 2006.

Adam Zagorecki, Mark Voortman and Marek J. Druzdzel. Decomposing local probability distributions in Bayesian networks for improved inference and parameter learning. In *Recent Advances in Artificial Intelligence: Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS-2006)*, Geoff Sutcliffe & Randy Goebel (eds), pages 860-865, Menlo Park, CA: AAAI Press, 2006.

Changhe Yuan and Marek J. Druzdzel. Importance sampling in Bayesian networks: An influence-based approximation strategy for importance functions. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 650-657, AUAI Press, Corvallis, OR, 2005.

Changhe Yuan and Marek J. Druzdzel. How heavy should the tails be? In *Recent Advances in Artificial Intelligence: Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS-2005)*, Ingrid Russell & Zdrawko Markov (eds), pages 799-804, Menlo Park, CA: AAAI Press, 2005.

Changhe Yuan, Tsai-Ching Lu and Marek J. Druzdzel (2004). Annealed MAP, In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 628-635, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2004.

Adam Zagorecki and Marek J. Druzdzel (2004). Elicitation of Probabilities under Noisy-OR Assumption. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS-2004)*, Ingrid Russell & John Kolen (eds), AAA Press, Key West, FL, pages 880-885, 2004.

Denver H. Dash and Marek J. Druzdzel (2003). A Robust Independence Test for Constraint-Based Learning of Causal Structure. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 167-174, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2003.

Changhe Yuan and Marek J. Druzdzel (2003). An Importance Sampling Algorithm Based on Evidence Pre-propagation. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 624-631, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2003.

Other peer reviewed conferences, symposia, workshops, and book chapters:

Adam Zagorecki and Marek J. Druzdzel. Probabilistic independence of causal influences. To appear in *Proceedings of The Third European Workshop on Probabilistic Graphical Models (PGM-06)*

Changhe Yuan and Marek J. Druzdzel. Hybrid loopy belief propagation, To appear in *Proceedings of The Third European Workshop on Probabilistic Graphical Models (PGM-06)*

Xiao Xun Sun, Marek J. Druzdzel and Changhe Yuan. Dynamic weighting A* search-based MAP algorithm for Bayesian networks. To appear in *Proceedings of The Third European Workshop on Probabilistic Graphical Models (PGM-06)*

Tsai-Ching Lu and Marek J. Druzdzel. Mechanism-based causal models for adaptive decision support. In *Challenges to Decision Support in a Changing World, Papers from the 2005 AAAI Spring Symposium*, Marek J. Druzdzel and Tze-Yun Leong (eds.), Technical Report SS-05-02, pages 73-79, Menlo Park, CA: AAAI Press, 2005

Daniel Garcia-Sanchez and Marek J. Druzdzel (2004). An Efficient Sampling Algorithm for Influence Diagrams. In Proceedings of the *Second European Workshop on Probabilistic Graphical Models (PGM-04)*, Peter Lucas (ed.), pages 97-104. Leiden, The Netherlands, October 2004.

Changhe Yuan and Marek J. Druzdzel (2004a). A Comparison of the Effectiveness of Two Heuristics for

Importance Sampling, In Proceedings of the *Second European Workshop on Probabilistic Graphical Models (PGM-04)*, Peter Lucas (ed.), pages 225-232. Leiden, The Netherlands, October 2004.

Agnieszka Onisko and Marek J. Druzdzel (2003). Effect of Imprecision in Probabilities on the Quality of Results in Bayesian Networks: An Empirical Study. In *Working Notes of the European Conference on Artificial Intelligence in Medicine (AIME-03) Workshop on Qualitative and Model-based Reasoning in Biomedicine*, pages 45-49, Protaras, Cyprus, 19 October, 2003.

F. Javier Diez, Marek J. Druzdzel and Miguel A. Hernan (2003). Causal diagrams to represent biases in the evaluation of diagnostic procedures. In *Proceedings of the 36th Annual Meeting of the Society for Epidemiologic Research (SER-03)*, Atlanta, GA, 2003.

Interactions / Transitions

a. Participation / presentations at meetings, conferences, seminars, etc.

(in addition to the conference presentations of papers listed in the publication list)

October 2005	Causal Graphs in Strategic Decision Making. United States Air Mobility Command, Scott Air Force Base, IL
Sept. 2005	Decision-analytic Methods in Medicine: Tools and their Usefulness. Philips Research, Briacliff Manor, NY
August 2005	There Are Good Reasons for SMILEing. New World Vistas AFOSR Progress Meeting, Saint Louis, MO
March 2005	Diagnostic Systems Based on Bayesian Networks: State of the Art and Some Directions for Further Work. Intel Research, Santa Clara, CA
March 2005	An Amazing Property of Joint Probability Distributions. Intelligent Systems Program's Artificial Intelligence Forum, University of Pittsburgh, Pittsburgh, PA
May 2004	Effect of Imprecision in Probabilities on the Quality of Results in Bayesian Networks: An Empirical Study. Department of Theoretical Physics, Institute of Physics, Uniwersytet Marii Curie Sklodowskiej (Maria Sklodowska Curie University), Lublin, Poland
April 2004	Importance Sampling Algorithms for Bayesian Networks: Principles and Performance. School of Computing, National University of Singapore
April 2004	Effect of Imprecision in Probabilities on the Quality of Results in Bayesian Networks: An Empirical Study. School of Computing, National University of Singapore
Sept. 2003	Effect of Imprecision in Probabilities on the Quality of Results in Bayesian Networks: An Empirical Study. Intelligent Systems Program's Artificial Intelligence Forum, University of Pittsburgh, Pittsburgh, PA
May 2003	More Good News About Importance Sampling in Bayesian Networks. New World Vistas AFOSR Progress Meeting, Estes Park, CO
April 2003	How Can Computers Improve Our Decision Making? North Boroughs Rotary International club meeting, Pittsburgh, Pennsylvania

b. Consultative and advisory functions to other laboratories

National Institute of Standards and Technology (NIST), Geithersburg, MD

United States Naval War College, Newport, RI

HRL Laboratories, Malibu, CA

c. Major fielding of our results

Here are some of the applications of our results and our software developed in the course of the project:

- Dr. John Lemmer (**John.Lemmer@rl.af.mil**) at the US Air Force Rome Laboratories uses our software and the results of our research on stochastic sampling algorithms in his work on causal analysis of dynamic military plans. The size of the problems that his group is facing is prohibitive and requires fast approximate algorithms. He has been incorporating our algorithms into jCAT, a tool fielded in military applications. The PI, Dr. Druzdzel, visited Rome Labs several times in the course of the project.
- Dr. Wojtek Przytula (**wojtek@hrl.com**) at the Hughes Raytheon Laboratories uses **GeNIe** and **SMILE**© in a diagnostic system for General Motors Diesel locomotives.
- Dr. Haiqin Wang (**haiqin.wang@boeing.com**) at Boeing is applying our software in her work on diagnosis. Another project using our software is collaboration between Boeing and Rockwell International, focusing on aircraft diagnosis.
- Researchers at Intel Research, Dr. John Mark Agosta (**john.m.agosta@intel.com**) and Dr. Thomas R. Gardos (**thomas.r.gardos@intel.com**) have incorporated **GeNIe** and **SMILE**© in their Smart Diagnostics project focusing on reducing the down time of expensive equipment used in manufacturing of integrated circuits. There is a good chance that this project will be ultimately fielded in Intel factories.
- Researchers at Philips Research, Dr. Kees van Zoon (**kees.van.zon@philips.com**) and Dr. Nicholas Chbat (**chbat@philips.com**) have been using **GeNIe** and **SMILE**© in their projects focusing on diagnosis of stroke.

Honors / Awards

- *2006 Catherine Ofiesh Orner Award* (with Adam Zagorecki and Mark Voortman) awarded school-wide for the best scholarly paper in Information Science submitted by a SIS student and co-authored by a SIS faculty member for the paper . “Decomposing local probability distributions in Bayesian networks for improved inference and parameter learning.”
- 2005 Honorable Mention in the *2005 IJCAII-JAIR Best Paper Prize* for the paper with Jian Cheng “AIS--BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks.” The IJCAII-JAIR Best Paper Prize is awarded to an outstanding paper published in JAIR in the preceding five calendar years. For the 2005 competition, papers published between 2000 and 2005 were eligible.
- *2004 Robert R. Korfhage Award* (with Adam Zagorecki), awarded school-wide for the best paper co-authored between a student and a faculty member, for the paper “How Common are Noisy-MAX Distributions in Practice?”
- *2005/2006 Andrew Mellon Graduate Fellowship* offered by the College of Arts and Sciences, University of Pittsburgh for Changhe Yuan, a doctoral student supported by the grant.
- *2004/2005 Andrew Mellon Graduate Fellowship* offered by the College of Arts and Sciences, University of Pittsburgh for Changhe Yuan, a doctoral student supported by the grant.